Simple and Fast Group Robustness by Automatic Feature Reweighting

Shikai Qiu [1 *] **Andres Potapczynski** [1 *] **Pavel Izmailov** [1] **Andrew Gordon Wilson** [1]

Abstract

A major challenge to out-of-distribution generalization is reliance on spurious features — patterns that are predictive of the class label in the training data distribution, but not causally related to the target. Standard methods for reducing the reliance on spurious features typically assume that we know what the spurious feature is, which is rarely true in the real world. Methods that attempt to alleviate this limitation are complex, hard to tune, and lead to a significant computational overhead compared to standard training. In this paper, we propose Automatic Feature Reweighting (AFR), an extremely simple and fast method for updating the model to reduce the reliance on spurious features. AFR retrains the last layer of a standard ERM-trained base model with a weighted loss that emphasizes the examples where the ERM model predicts poorly, automatically upweighting the minority group without group labels. With this simple procedure, we improve upon the best reported results among competing methods trained without spurious attributes on several vision and natural language classification benchmarks, using only a fraction of their compute.

1. Introduction

Most realistic datasets contain features that can be used to achieve strong performance in-distribution, but that are not inherently relevant to the predictive task. For example, on Waterbirds (Sagawa et al., 2020), an image classification dataset where the goal is to distinguish landbirds (e.g. woodpecker) from waterbirds (e.g. seagull), a model can reach 95% in-distribution accuracy by only looking at the background and completely ignoring the actual bird.

[*]Equal contribution [1]New York University. Correspondence to: Shikai Qiu <sq2129@nyu.edu>, Andres Potapczynski <ap6604@nyu.edu>, Pavel Izmailov <pi390@nyu.edu>, Andrew Gordon Wilson <andrewgw@cims.nyu.edu>.

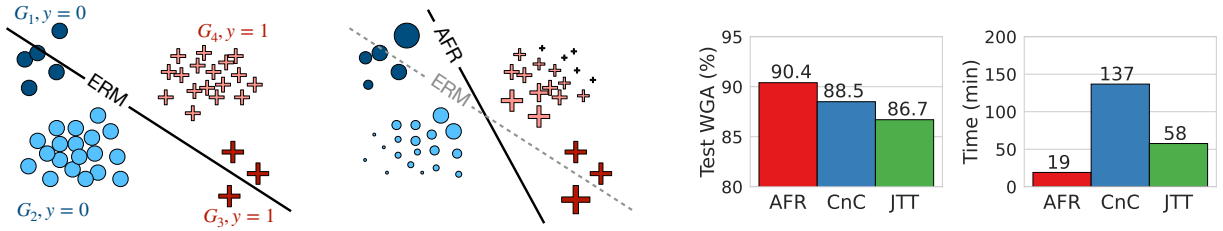
Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

Features such as background (ocean), which are correlated with but not causally related to the target (waterbird) are known as *spurious features* or *spurious attributes*. By relying on spurious features, a model performs well on the training data distribution without learning the *core features* that are intrinsically descriptive of the targets. Such models inevitably fail to generalize to new data where the spurious correlation breaks, such as images of waterbirds appearing on land backgrounds.

The best existing approaches for addressing spurious features require access to group labels on the training data (Sagawa et al., 2020; Idrissi et al., 2021), which simultaneously specify both the class label and the spurious attribute for each datapoint. This requirement presents a major limitation: while many real-world problems contain spurious correlations, we do not know what they are a priori! Moreover, even in rare cases where we could potentially identify the spurious feature, it can be prohibitively expensive to manually add group labels to a large dataset. Consequently, there has been great interest in reducing the reliance on spurious features without explicit access to the spurious attributes (e.g., Liu et al., 2021; Nam et al., 2020; Zhang et al., 2022; Lee et al., 2022b). However, compared to standard training, the resulting methods are generally more complex, computationally heavy, and difficult to tune, making them difficult to adopt in practice.

In this paper, we propose *Automatic Feature Reweighting* (AFR), a simple and fast method for reducing the reliance on the spurious attributes with a minimal computational overhead compared to standard training. As in Kirichenko et al. (2022), we freeze the feature extractor of the base model pretrained on a dataset with a spurious correlation, and focus on retraining the last layer of the model. However, unlike Kirichenko et al. (2022), which uses a group-balanced dataset for re-training, we use an weighted loss on a held-out dataset *drawn simply from the training distribution* for retraining the last layer, where the weights prioritize datapoints on which the base model performs poorly. These weights automatically result in an approximately group-balanced dataset for re-training without either group labels or intervention in base model training. We illustrate AFR in Figure 1.

We show that AFR achieves strong results on vision and lan-



(a) ERM classifier (b) AFR classifier (c) Performance and training time on Waterbirds

Figure 1: **Automatic Feature Reweighting. (a)** An illustration of the ERM classifier trained on a binary classification problem with a spurious correlation. The dataset consists of four groups shown in different colours, and two classes shown with circles and pluses. The ERM classifier (decision boundary shown with the black line) performs well on the majority groups G_2, G_4 (shown in lighter colors), but underperforms on the minority groups G_1, G_3 (lighter colors). **(b)** AFR upweights the datapoints where the base ERM model achieves high loss. The size of a marker reflects the weight of the corresponding datapoint. This weighting automatically prioritizes the minority data, leading to a classifier that achieves much better performance on the minority groups. **(c)** Test worst-group accuracy and training time comparison on Waterbirds for state-of-the-art methods that do not use group information during training. AFR outperforms the baselines, while only requiring a small fraction of compute time.

guage benchmarks such as Waterbirds, CelebA, MultiNLI, CivilComments and Chest X-Ray while requiring a negligible training time overhead compared to ERM. In particular, we improve upon the best reported results on Waterbirds and MultiNLI among methods which do not require labels for the spurious attributes during training. Furthermore, we show that AFR can significantly outperform a state-of-the-art method that explicitly uses group labels during training, when only a small number of group labels are available. Through extensive ablations, we find AFR is robust to its hyperparameters such that it often significantly improves group robustness without careful hyperparameter tuning.

Code for AFR is available at
`https://github.com/AndPotap/afr`.

2. Preliminaries

We frame the group robustness problem and outline several baseline methods.

2.1. Problem Setting

We consider the group robustness setting (Sagawa et al., 2020). Specifically, we assume that the data distribution consists of several *groups* $g \in \mathcal{G}$, which are typically defined by a combination of the class label $y \in \mathcal{Y}$ and a spurious attribute $s \in \mathcal{S}$. For example, in CelebA where we classify $y \in \{\text{Blond}, \text{Not Blond}\}$, the spurious attribute is the gender of the person shown in the image $s \in \{\text{Male}, \text{Female}\}$. The groups then correspond to all pairs of values of the class label and spurious attribute $\mathcal{G} = \mathcal{Y} \times \mathcal{S}$.

The attribute s is spurious if it is correlated with y, but not causally related with it. In CelebA, about 94% of the datapoints with $y = \text{Blond}$ have the spurious attribute $s = \text{Female}$. As a result, models trained on this problem rely on the gender feature to predict the hair color, and underperform on the minority group $g = (\text{Blond}, \text{Male})$. Throughout the paper, we will be evaluating the *worst group accuracy* (WGA), i.e. the minimum of predictive accuracies of our model across all groups.

Access to spurious attributes. Most existing methods for robustness to spurious correlations assume access to spurious attributes s on the training data (Sagawa et al., 2020), or on a subset of the available data used to train the parameters of the model (Kirichenko et al., 2022; Nam et al., 2022; Sohoni et al., 2021). In contrast, we consider the more challenging setting, where the group attributes are not available to us on any of the datapoints used to train the parameters of the model. We underscore that, as all other methods following this setting (Liu et al., 2021; Zhang et al., 2022), we tune hyperparameters on a validation set with explicit spurious attributes. However, as seen in Section 4.4, our method does not require a large amount of validation data.

2.2. Group Robustness Baselines

Empirical Risk Minimization (ERM). In ERM, we learn the parameters θ of the model by minimizing the loss ℓ averaged over the training data:

$$\mathcal{L}^{\text{ERM}}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \ell(x_i, y_i; \theta). \tag{1}$$

We use cross-entropy $\ell^{\text{xe}}(x_i, y_i; \theta) = -\log p_{y_i}(x_i; \theta)$, where $p_y(x; \theta)$ is the probability of the class y predicted by the model with parameters θ on an input x.

Group Distributionally Robust Optimization (GDRO). Intuitively, GDRO (Sagawa et al., 2020) minimizes the worst group loss given by

$$\mathcal{L}^{\text{GDRO}}(\theta) = \max_{g \in \mathcal{G}} \frac{1}{N_g} \sum_{i:g_i=g} \ell(x_i, y_i; \theta)$$

where N_g is the number of observations with group $g_i = g$. To compute the GDRO objective, we require group labels g_n for all of the training data and these group labels help GDRO to considerably improve worst group performance. Therefore, the worst group accuracy achieved by GDRO is considered an upper bound on a given dataset. Sagawa et al. (2020) provides full details of the method.

Deep Feature Reweighting (DFR). Kirichenko et al. (2022) showed that it is sufficient to retrain just the last layer of a classifier on a group-balanced held-out *reweighting* dataset to obtain results similar to GDRO. Specifically, they represent the model $m_\theta = c_\phi \circ e_\psi$, where e_ψ is the feature extractor with parameters ψ and c_ϕ is the classification head (last layer) with parameters ϕ, and $\theta = (\phi, \psi)$. Starting with a pretrained model, we freeze the feature extractor parameters ψ, and retrain only the classifier c_ϕ using a held-out dataset (not used to train the feature extractor) where the number of datapoints from each group is the same. They use the ERM objective in Eq. (1) to retrain the classifier. While DFR does not require group annotations on all of the training data like GDRO, it requires a group-balanced reweighting dataset.

Just Train Twice (JTT). In JTT (Liu et al., 2021), we first train a classifier m_θ with ERM for T steps finding weights $\hat{\theta}$, and then construct the set $\mathcal{E} = \{i : \arg\max_y m_{\hat{\theta}}(x_i)[y] \neq y_i\}$ of misclassified training indices. We then retrain m_θ by minimizing the loss

$$\mathcal{L}^{\text{JTT}}(\theta) = \frac{\lambda}{|\mathcal{E}|} \sum_{i \in \mathcal{E}} \ell^{\text{xe}}(x_i, y_i; \theta) + \frac{1}{|\mathcal{E}^c|} \sum_{i \notin \mathcal{E}} \ell^{\text{xe}}(x_i, y_i; \theta),$$

where the hyperparameter $\lambda \in \mathbb{R}$ upweights the relevance of the missclassified observations. JTT does not explicitly require access to the group labels. Importantly, JTT tunes the number T of epochs for which the first model is trained to avoid both under and over-fitting the training set. If T is too small, the error set will approach the entire training set. If T is too large, the error set will approach the empty set.

Correct-n-Contrast (CNC). The method in Zhang et al. (2022) builds upon JTT and also trains two models, but it uses a contrastive loss to train the second model. The first model is used to define the positive pairs (objects with the same class, but different predictions), and negative pairs (objects with different classes but the same prediction). CNC achieves strong WGA without requiring group annotations, but requires tuning and comes at a large computational overhead compared to standard training: the authors report $10\times$ training time for CNC compared to ERM on multiple datasets. Similar to how JTT requires tuning the number of epochs to train the first model, CNC requires tuning the weight decay parameter for training the first model to avoid both under and over-fitting the training set in order to extract information on the spurious attributes from its prediction.

We provide an extended discussion of additional related work in Appendix D.

Algorithm 1 Automatic Feature Reweighting

Input: Training set partitioned into $\mathcal{D}_{\text{ERM}}$ (80%), $\mathcal{D}_{\text{RW}}$ (20%), $\gamma \geq 0$ and a classifier that decomposes as $m_\theta = c_\phi \circ e_\psi$ where $\theta = (\phi, \psi)$.
Stage 1: Train checkpoint $\hat{\theta} = (\hat{\phi}, \hat{\psi})$ until convergence on $\mathcal{D}_{\text{ERM}}$ using the loss $\mathcal{L}^{\text{ERM}}$.
Stage 2: Re-train last layer c_ϕ on $\mathcal{D}_{\text{RW}}$ using the loss $\mathcal{L}^{\text{AFR}}$ in Eq. (2), leaving the feature extractor $e_{\hat{\psi}}$ fixed.

3. Automatic Feature Reweighting

We now introduce AFR, a fast and simple method to improve group robustness without group annotations during training. As we will show, AFR is based on the key insight that we can infer minority group examples from an ERM model alone, trained in the standard way without intervention. The ability to use a standard ERM model is an important advantage over JTT and CNC where a model is trained specifically to learn the spurious features by applying heavy regularization. We use the inferred information about the minority examples to automatically construct a more group-balanced dataset and retrain only the last layer of the ERM model without access to any group labels, unlike DFR where group labels are required.

3.1. Method Description

At a high level, AFR proceeds in two stages. In the first stage, we train an ERM model on the training set $\mathcal{D}_{\text{ERM}}$. In the second stage, we retrain only the last layer of the model on a reweighting set $\mathcal{D}_{\text{RW}}$, with examples receiving high loss under the ERM checkpoint upweighted. Here, $\mathcal{D}_{\text{ERM}}$ and $\mathcal{D}_{\text{RW}}$ are two distinct datasets drawn from the same distribution and neither of them needs to be group-balanced. We find that splitting the training set in a 80%–20% proportion to construct $\mathcal{D}_{\text{ERM}}$ and $\mathcal{D}_{\text{RW}}$ works well in practice, but we show that performance is not particularly sensitive to the split in Appendix C.6.

We summarize AFR in Algorithm 1. We now describe the two stages in further detail.

Stage 1. We train a model checkpoint on $\mathcal{D}_{\text{ERM}}$ *until*

convergence via standard ERM without any modifications to the standard training procedure. Typically in this stage, we would achieve near 100% accuracy on the training data and the resulting model would have poor test performance on minority groups. In contrast to JTT and CNC (see Section 2.2), we do not under-trained our first stage model to avoid completely fitting the training data, i.e., achieving near-zero training cross-entropy loss.

Stage 2. Denote the model parameters as $\theta = (\phi, \psi)$, where ϕ represents the last layer parameters and ψ represents all the other parameters and denote by $\hat{\theta} = (\hat{\phi}, \hat{\psi})$ their values learned at the end of stage 1. In the second stage, we retrain the last layer parameters ϕ of the model on $\mathcal{D}_{\text{RW}}$ using the following objective:

$$\mathcal{L}^{\text{AFR}}(\phi) = \sum_{i=1}^{M} \mu_i \ell^{\text{xe}}\left(x_i, y_i; \phi, \hat{\psi}\right) + \lambda \left\|\phi - \hat{\phi}\right\|_2^2, \quad (2)$$

where $M = |\mathcal{D}_{\text{RW}}|$, and $\{\mu_i\}_{i=1}^M$ are per-example weights for the cross-entropy loss ℓ^{xe}. These weights are designed to be large for datapoints where the original model $\hat{\theta}$ predicts poorly, thus automatically upweighting the minority groups. We consider a simple functional form for the weights given by a softmax over the per-example "incorrectness" $1 - \hat{p}_i$, where $\hat{p}_i$ is the probability for the correct class y_i assigned by the stage 1 checkpoint $\hat{\theta}$, with a tunable inverse temperature $\gamma \geq 0$: $\mu_i \propto \exp(\gamma\,(1 - \hat{p}_i)) \propto \exp(-\gamma\,\hat{p}_i)$. To address datasets with class imbalance, we further modulate the per-example weight with a class-dependent constant and define

$$\mu_i = \frac{\beta_{y_i} \exp(-\gamma\,\hat{p}_i)}{\sum_{j=1}^{M} \beta_{y_j} \exp(-\gamma\,\hat{p}_j)}, \quad (3)$$

where β_y is one divided by the number of examples belonging to class y in the reweighting set. Note that the weights μ_i in Eq. (3) are only computed once and fixed during the second stage.

The hyperparameter γ specifies how much to upweight examples with poor predictions, while $\lambda \geq 0$ is a regularization hyperparameter to prevent the last layer from focusing only on minority examples at the expense of degrading performance on majority group examples to an unacceptable level. The regularization also reduces overfitting to limited reweighting data. In Section 5, we investigate how optimal our weighting function is for group robustness.

Effect of γ. The hyperparameter γ determines how strongly we upweight the datapoints where the first stage model $\hat{\theta}$ provides poor predictions. In particular, by setting $\gamma = 0$, we recover standard ERM (with class reweighting), as all the weights in Eq. (3) become simply $\mu_i \propto \beta_{y_i}$. In contrast, as we increase γ, AFR's loss concentrates more on the poorly classified points.

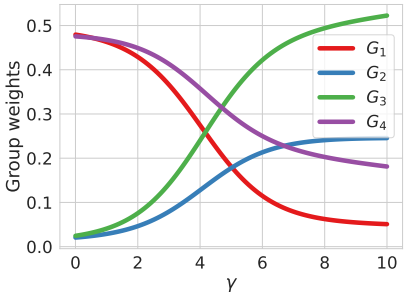


Figure 2: **AFR weights on Waterbirds.** Group aggregated weights as a function of γ on $\mathcal{D}_{\text{RW}}$ for the Waterbirds dataset, with majority groups G_1 and G_4, and minority groups G_1 and G_2. The ERM model performs poorly on minority groups G_2 and G_3, enabling automatic group rebalancing by upweighting poorly predicted examples. The reweighted group distribution is more balanced than the original group distribution, for a broad range of γ values.

To illustrate how we automatically upweight minority examples and reduce group imbalance with the weights $\{\mu_i\}_{i=1}^M$, in Figure 2 we plot the group aggregated weights defined as the sum $\sum_{i:g_i=g} \mu_i$ of all per-example weights within each group g, as a function of γ on $\mathcal{D}_{\text{RW}}$ for the Waterbirds dataset. As we increase γ, the minority groups (G_2 and G_3) receive increasingly higher weights. The re-weighted group distribution is more balanced than the original group distribution, regardless of the value of γ, as long as it is positive. At $\gamma = 4.4$, the group aggregated weights are $23\%, 15\%, 31\%$ and 31%, near the ideal scenario of 25% each.

In the following sections, we expand on key design decisions in AFR and important conceptual differences between AFR and existing work that underlies AFR's strong performance, simplicity, and efficiency.

3.2. Inferring Minority Examples without Intervention

One key insight that distinguishes AFR from JTT and CNC is that we do not need to intervene on the ERM model (e.g. by over-regularizing) to infer spurious attributes or to identify minority group examples. While an ERM model trained via the standard procedure can fit all datapoints in its training set $\mathcal{D}_{\text{ERM}}$, its performance on minority group examples in a held-out dataset will be much worse compared to the majority group examples. Indeed, this performance discrepancy is precisely why spurious correlations are problematic in the first space. In AFR, we take advantage of this performance discrepancy to automatically isolate and upweight the minority group examples on the reweighting set $\mathcal{D}_{\text{RW}}$ during the second stage to retrain only the last layer.

In contrast, both JTT and CNC rely on heavy regularization

to encourage the model to under-fit the data and only learn spurious features in the first stage. These methods are thus more expensive and harder to tune, as the entire model has to be retrained from scratch in order to find, for example, the optimal regularization strength. Moreover, an entirely new model needs to be trained in the second stage to learn the core features relevant to the actual classification problem, resulting in substantial computational overhead over training a single ERM model.

A potential downside of splitting the training data into $\mathcal{D}_{\text{ERM}}$ and $\mathcal{D}_{\text{RW}}$ is that we only use a subset of the available data for training the feature extractor. As such, we may learn a slightly weaker feature extractor compared to a standard ERM model trained with the entire training set. However, by reweighting the last layer with the remaining data $\mathcal{D}_{\text{RW}}$, AFR is able to achieve much higher worst group accuracies than ERM, showing that the benefit of the procedure drastically outweighs its cost when the goal is to achieve group robustness. In Appendix C.6 we also show AFR is not very sensitive to the choice of this splitting ratio.

3.3. Last Layer Retraining without Group Labels

Retraining only the last layer on the reweighting set is an important design decision in AFR that has several important benefits. First, as a standard ERM model learns the core features sufficiently well despite spurious correlations, retraining only the last layer is sufficient to significantly improve group robustness (Kirichenko et al., 2022; Lee et al., 2022a). Second, as the last layer tends to have several orders of magnitude fewer parameters than the entire model, retraining only the last layer is more data efficient than retraining the entire model. This data efficiency enables us to use a small reweighting set and leave most of the training data to train the feature extractor during the first stage. Finally, retraining the last layer is extremely fast. By caching the embeddings from the fixed feature extractor, we can complete the last layer retraining in less than a minute on standard hardware.

In contrast to DFR, our reweighting set does not need to be group-balanced and can simply be drawn from the training distribution. The weighting in Eq. (3) automatically constructs a more balanced dataset. As such, AFR can tackle problems where no group-annotated data exists, whereas DFR cannot.

3.4. AFR's Advantages over Alternatives

To the best of our knowledge, AFR is the only group robustness method to achieve state-of-the-art results across a wide variety of benchmarks without requiring group labels through training, and with negligible computational overhead over standard ERM, as we will show in Section 4. We summarize AFR's advantages over prior group robustness methods as follows:

- **More broadly applicable:** Unlike methods such as DFR and GDRO, AFR does not require group labels during training and is therefore applicable to problems where DFR and GDRO are not due to unavailability of group annotated data.
- **Simpler to use:** To run AFR, we only need a standard ERM checkpoint, potentially from an existing pretrained model. We then only retrain its last layer using a weighted cross-entropy loss and ℓ_2 regularization to the initial weights. Unlike JTT and CNC, AFR does not require any intervention for the first stage of training, or implementing special training objectives and batch samplers for the second stage.
- **Faster to train:** AFR only retrains the last layer parameters, often taking less than a minute by caching the embeddings $\{e_{\hat{\psi}}(x)\}_{x\in\mathcal{D}_{\text{RW}}}$, while JTT and CNC train a second model entirely from scratch, which can take hours. We show a comparison of training times in Figure 3.
- **Easier to tune:** Since AFR has no hyperparameter for training the first checkpoint, unlike JTT and CNC, hyperparameter tuning is much easier and cheaper for AFR. Indeed, as a single first stage checkpoint can be reused, the time for sweeping over K hyperparameter settings is roughly $O(1)$ for AFR but $O(K)$ for JTT and CNC since training time for the last layer is negligible compared to retraining the entire model.

These advantages reflect important conceptual differences between AFR and alternatives as outlined in previous sections, despite a few high level similarities to methods like JTT and DFR, such as two-stage training.

4. Experiments

We evaluate AFR on a range of benchmarks and provide detailed ablations on design decisions and hyperparameters.

4.1. Datasets, Models and Hyperparameters

We now describe the datasets, models and hyperparameters that we use in the experiments.

Datasets. We consider several image and text classification problems. For more details, see Appendix B.

- **Waterbirds** (Sagawa et al., 2020) is an image classification dataset where the goal is to classify birds into landbirds (woodpecker, catbird, etc) and waterbirds (albatross, seagull, etc). The background is a spurious feature, as most waterbirds are shown on water backgrounds and most landbirds are shown on land.

Method	Waterbirds		CelebA		MultiNLI		CivilComments		Chest X-Ray	
	Worst(%)	Mean(%)	Worst(%)	Mean(%)	Worst(%)	Mean(%)	Worst(%)	Mean(%)	Worst(%)	Mean(%)
ERM	72.6	97.3	47.2	95.6	67.9	82.4	57.4	92.6	4.4	95.3
JTT	86.7	93.3	81.1	88.0	72.6	78.6	**69.3**	**91.1**	52.3	64.2
CNC	$88.5_{\pm 0.3}$	$90.9_{\pm 0.1}$	$\mathbf{88.8_{\pm 0.9}}$	$89.9_{\pm 0.5}$	–	–	$68.9_{\pm 2.1}$	$81.7_{\pm 0.5}$	–	–
AFR (Ours)	$\mathbf{90.4_{\pm 1.1}}$	$\mathbf{94.2_{\pm 1.2}}$	$82.0_{\pm 0.5}$	$\mathbf{91.3_{\pm 0.3}}$	$\mathbf{73.4_{\pm 0.6}}$	$\mathbf{81.4_{\pm 0.2}}$	$68.7_{\pm 0.6}$	$89.8_{\pm 0.6}$	$\mathbf{56.0_{\pm 3.4}}$	$\mathbf{70.1_{\pm 6.0}}$
Group-DRO†	91.4	93.5	88.9	92.9	77.7	81.4	69.9	88.9	–	–
DFR†	$92.9_{\pm 0.2}$	$94.2_{\pm 0.4}$	$88.3_{\pm 1.1}$	$91.3_{\pm 0.3}$	$74.7_{\pm 0.7}$	$82.1_{\pm 0.2}$	$70.1_{\pm 0.8}$	$87.2_{\pm 0.3}$	$59.8_{\pm 1.8}$	$64.2_{\pm 3.1}$

Table 1: **Results on spurious correlation benchmarks.** We report test worst-group accuracy and test mean accuracy. Additionally, † denotes oracle methods that make explicit use of group annotations. For AFR, we report the mean ± std over 3 independent runs. We report CNC numbers from Zhang et al. (2022), ERM and JTT numbers from Liu et al. (2021) and DFR numbers from Kirichenko et al. (2022). For Chest X-Ray, we report the JTT number from Yang et al. (2022) and the ERM number from our own run. For mean accuracy, we follow Liu et al. (2015) and Sagawa et al. (2020) and weight the group accuracies according to their prevalence in the training data. AFR provides competitive results with substantially lower runtime (shown in Figure 3) than alternatives that do not use group information during training.

- **CelebA** (Liu et al., 2015) is an image classification dataset, where we focus on classifying whether an image shows a person with blond hair or not. Gender is the spurious feature, as 94% of images showing people with blond hair in CelebA are of women, and blond men constitute a minority group.
- **MultiNLI** (Williams et al., 2017) is a text classification problem, where we discern whether the second sentence in a given pair of sentences is entailed by, contradicts or is neutral to the first sentence. The spurious attribute is the presence of negation words such as "never" which correlates with "contradiction".
- **CivilComments** (Borkan et al., 2019) is a text classification benchmark, where we classify an online comment as "toxic" or "not toxic". We use the version from the WILDS benchmark (Koh et al., 2021), where the presence of text related to gender (male, female), sexual orientation (LGBTQ), race (black, white) and religion (Christian, Muslim or other) is spuriously correlated with the comment being labeled as "toxic".
- **CXR** (Yang et al., 2022) is an image classification dataset where we consider the task of predicting pneumonia based on chest X-rays from two sources: CheXpert (Irvin et al., 2019) and NIH (Wang et al., 2017). Here the spurious feature is the source of the image (machine-specific artifacts).

Models. We follow standard model choices consistent with the baselines on each of the datasets (Liu et al., 2021; Kirichenko et al., 2022; Zhang et al., 2022): for Waterbirds and CelebA we use ResNet-50 (He et al., 2016) pretrained on ImageNet1k (Russakovsky et al., 2015), for MultiNLI and CivilComments we use BERT (Devlin et al., 2018) pre-trained on Book Corpus and English Wikipedia data, and for Chest X-Ray we use DenseNet-121 (pretrained on ImageNet1K). We provide further details in Appendix B.

Hyperparameters. Following all other group robustness methods, we use validation WGA to tune AFR's γ and λ and perform early stopping (see Appendix B for details).

4.2. AFR Achieves Strong Group Robustness

We report the results for AFR and the baselines in Table 1, showing that AFR outperforms or that it is competitive with the best reported results among methods trained without access to the spurious attributes on all datasets except CelebA. AFR improves upon the best reported results on Waterbirds and MultiNLI. Compared to JTT, AFR achieves better performance on all datasets except for CivilComments where the results are slightly worse. Compared to CNC, AFR achieves better results on Waterbirds, similar results on CivilComments, and underperforms on CelebA. In Appendix C.1 we present additional experiments where we show that AFR can be safely applied to data without known spurious features without losing performance.

4.3. AFR is Much Faster Than Alternatives

In Figure 3, we visualize the training time required by AFR and baselines on each of the datasets (see Appendix B.2 for more details). Unsurprisingly, AFR incurs a negligible overhead (6% on average) compared to ERM by only retraining the last layer. Here we included the time to pre-compute and cache the embeddings, which in practice can be ignored by amortizing across a hyperparameter sweep. In contrast, JTT is on average three times as expensive (205% overhead compared to ERM). It might appear counter-intuitive that JTT takes 3× the time of ERM, as we are "just training twice"! However, both stages of JTT, especially the last one, are more expensive than ERM. First, the initial checkpoint

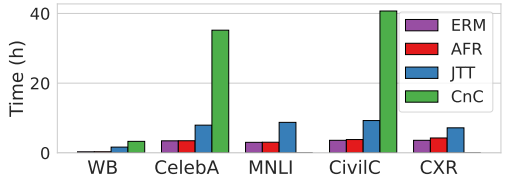


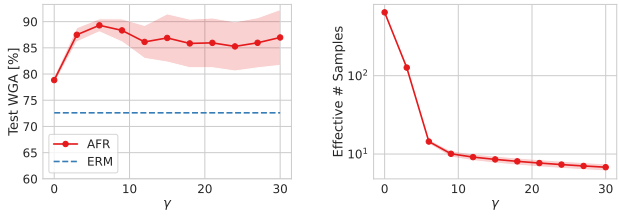
Figure 3: **Training time.** Training time (in hours) across different spurious correlation benchmarks. AFR only incurs a negligible overhead compared to standard ERM and is substantially less expensive than JTT and CNC.

takes longer to run as it converges slower due to high regularization (training to convergence is needed in order to tune the first stage epoch number). Second, the second phase has substantially more data which increases the epoch runtime, as all misclassified examples are repeated multiple times in the dataset. Finally, CNC is the least scalable method with an average overhead of 550% compared to ERM. CNC shares similar steps to JTT but the contrastive objective is expensive to optimize.

4.4. AFR is Group Label Efficient

Efficiency in terms of group labels is valuable for group robustness methods, because correctly identifying the spurious attributes and collecting corresponding group labels can be a time-consuming and expensive process. Unlike methods such as DFR and Group DRO, AFR can be applied when group labels are not available on the training set. On the other hand, while AFR is generally not sensitive to the choice of hyperparameters, as we show in Section 4.5 and Appendix C, optimal performance still requires hyperparameter tuning on a group-annotated validation set, and the same applies to JTT and CNC. However, since AFR only uses group labels for hyperparameter tuning and not for training, it is reasonable to expect AFR to be more efficient in terms of group labels than methods like DFR. In Figure 5, we show that AFR can indeed achieve significantly higher test WGA than DFR when only a small fraction of group-annotated validation data are accessible to both methods. On Waterbirds, for example, AFR achieves near-maximal test WGA improvement with only 0.5% of the group-annotated validation data, consisting of merely 5 examples, whereas DFR achieves lower test WGA than ERM by overfitting to these examples. On the other hand, since DFR directly uses group labels to train the model, rather than only to tune the hyperparameters, it can eventually outperform AFR as more group labels become available, as shown in the case of CelebA.

As a result, we conclude that AFR is often more efficient in terms of group labels than DFR when the number of available group labels is limited, though not necessarily when group labels are abundant.



(a) Test WGA (b) Effective sample size

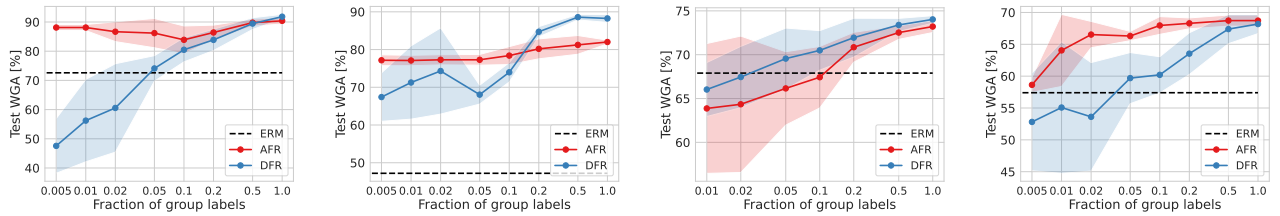
Figure 4: **Robustness to γ. (a)** AFR significantly improves test WGA on Waterbirds for any $\gamma \in [0, 30]$. We show mean and standard deviation (error bars) across 3 runs. **(b)** The effective sample size decreases and then stabilizes as γ grows, explaining stability in AFR's performance as γ increases.

4.5. AFR is Robust to Hyperparameters

We study the robustness of AFR to the choice of the hyperparameter γ for Waterbirds. Furthermore, in Appendix C, we present additional results ablating early stopping, ℓ_2 regularization to initial checkpoint, dataset splitting ratio, and the functional form of the weights, demonstrating AFR is also robust to these hyperparameters and design decisions.

As argued earlier, we do not expect the performance to be highly sensitive to γ, since as long as γ is positive, the reweighted group distribution will be more balanced than the original group distribution. We confirm this intuition in Figure 4(a), which shows the mean and standard deviation of the test worst group accuracy achieved by our method over 3 runs on Waterbirds across a wide range of γ, keeping $\lambda = 0$ for simplicity. AFR achieves near-optimal test WGA by the time $\gamma = 6$. As we continue to increase γ, the test WGA degrades slightly and then stabilizes around a value still much higher than ERM. To understand this behavior, note that for a sufficiently large γ, the set of examples that receive non-negligible weights μ_i converges approximately to a fixed set, namely, the set of most poorly predicted examples under the first stage checkpoint. This property is evident in Figure 4(b), where we show the effective sample size N_{eff} approximately stabilizes after initially plummeting for $\gamma < 6$. Here, $N_{\text{eff}} = 1/\sum_{i=1}^{M} \mu_i^2$ represents the effective number of observations for weighted samples (Kish, 1965). As a result, the performance of AFR doesn't vary significantly beyond $\gamma = 6$, revealing only a gradual decrease in mean and an increase in variance as the effective sample size slowly diminishes. Overall, AFR maintains a significant improvement over ERM for all $\gamma \in [0, 30]$. In Appendix C.2, we show AFR is even more robust to γ on CelebA where its performance is almost constant over γ.

Setting $\gamma = 0$ already yields a significant increase in test WGA over standard ERM. This improvement is not surpris-



(a) Waterbirds (b) CelebA (c) MultiNLI (d) CivilComments

Figure 5: **Group label efficiency of AFR vs DFR.** AFR significantly outperforms DFR on 3 out of 4 datasets when only a small fraction of group-annotated validation data is used by AFR and DFR for hyperparameter tuning and training respectively. We show means and standard deviations computed over 9 runs, with 3 seeds for base model training and 3 seeds for validation set down-sampling.

ing since the per-example weights μ_i in AFR account for class imbalance and the smallest group in Waterbirds indeed belongs to the less represented class.

5. Are AFR Weights Optimal?

In this section, we explore how optimal the AFR weights are for improving group robustness. We find that it is not always possible for AFR to produce group-balanced weights and we could achieve better performance if we had access to such weights. However, we also find that AFR's weights are in fact *near-optimal* for maximally re-balancing the group distribution, given constraints on the available information and practical considerations. These findings suggest both that AFR should be very effective among approaches based on dataset reweighting, and that these approaches, including AFR and JTT, share intrinsic limitations, and alternative approaches, e.g. CNC, may provide better results on certain datasets, such as CelebA.

5.1. AFR Weights Are Imbalanced on CelebA

Kirichenko et al. (2022) showed that retraining the last layer of an ERM checkpoint with a group-balanced held-out set is sufficient for state-of-the-art worst group accuracy. By upweighting minority group examples, AFR aims to automatically construct a group-balanced held-out set without access to group labels. However, achieving completely balanced group weights with AFR is not always possible.

To illustrate this point, we analyze how AFR weights distribute over different groups, and study their training dynamics, as a function of γ on CelebA, where AFR does not match state of the art results. In Figure 6, we show the group aggregated weights produced by AFR on CelebA and the training accuracy per group during the second stage. While AFR correctly upweights the two groups G_4 and G_3 with the lowest accuracy and downweights the remaining groups G_1 and G_2, at no value of γ are the group aggregated weights close to balanced, unlike on Waterbirds (see Figure 2). Either the worst group G_4 is assigned too small a weight that it is not sufficiently optimized for by the model as in Figure 6(b), or the second worst group G_3 is assigned too large a weight that the accuracy on one of the other groups (G_1) drops significantly as seen in Figure 6(c). The reason AFR assigns a much larger weight to G_3 than to G_4 is because G_3 is 14 times larger in population than G_4, despite being the second worst group.

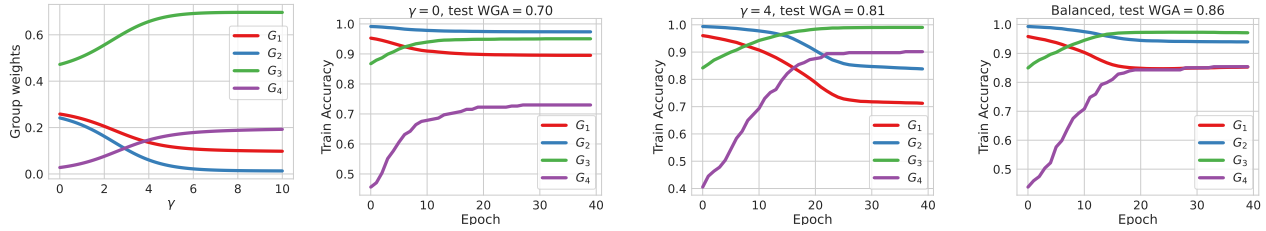
To demonstrate that the imbalance of the weights is the cause for AFR's sub-optimal performance, we show in Figure 6(d) that last layer retraining on the reweighting set can achieve as high as 86% train and test WGA if we had access to group-balanced weights where $\mu_i \propto 1/|G_i|$ with $|G_i|$ denoting the size of group i.

5.2. AFR Weights are Near-Optimal Given Constraints

AFR weights are not optimal in the unrestricted sense, since group-balanced weights perform better on CelebA. But are AFR weights optimal without using additional information?

In Appendix C.7, we present strong evidence showing that, rather than AFR's specific choice of $\mu_i \propto \beta_{y_i} \exp(-\gamma\, \hat{p}_i)$ being ineffective, there in fact exists no function of only the first-stage predictions (which contains $\hat{p}_i$) and the true class label y_i that can produce completely balanced weights on CelebA. We do this by showing a neural network trained to maximally balance group weights is unable to do so given the first-stage predictions and class labels. We provide training details in Appendix C.7. This result indicates that, without access to additional information, any approach based on reweighting the dataset, such as AFR and JTT, is unlikely to lead to optimal performance on CelebA, potentially explaining the fact that both AFR and JTT are outperformed by CNC on this dataset by a wide margin.

In Figure 7, we plot the learned weighting function against $\hat{p}$ for the each class on Waterbirds and CelebA. Using a log-scale for the y-axis, all learned functions follow an approximately straight line, showing that the optimal weighting

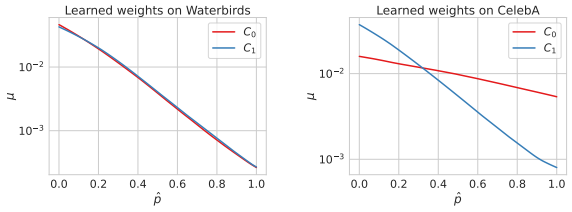


(a) Group aggreagated weights (b) $\gamma = 0$ (c) $\gamma = 4$ (d) Group-balanced

Figure 6: **AFR on CelebA. (a)** AFR upweights the minority groups but does not achieve group-balance. **(b)** $\gamma = 0$: AFR weight on the worst group G_4 is not high enough. **(c)** $\gamma = 4$: Accuracy increases for G_3 and G_4 but decreases for G_1 and G_2 due to group-imbalance in AFR weights. **(d)** Group-balanced weights improves training dynamics and performance.

function is indeed well-modelled by a decaying exponential in $\hat{p}$. Furthermore, the class with fewer examples (C_1) is offset by a positive constant in log-scale from the other class (C_0) in CelebA, showing that it is useful to allow a class-dependent multiplicative constant before the exponential for addressing class imbalance.

Since neural networks can learn functions much more complex than what can be represented with a single parameter γ, unsurprisingly there are still some differences between the learned functional form and ours: in log-space the weights aren't perfectly linear as a function of $\hat{p}$ and on CelebA the slope, which corresponds to $-\gamma$ in AFR, is class-dependent, which can be accounted for at the cost of more hyperparameters. Given the constraints that 1) we can only afford a handful of hyperparameters so that cross-validation with a small group-annotated validation set is effective and efficient, and 2) no additional information, such as group labels, is used to define the weights, AFR's specification of the weights, $\mu_i \propto \beta_{y_i} \exp(-\gamma\, \hat{p}_i)$, is in fact near-optimal in maximally re-balancing the group distribution.



(a) Waterbirds (b) CelebA

Figure 7: The optimal weighting function approximated by a neural network resembles the one used by AFR: a decaying exponential in the predicted probability $\hat{p}$ for the correct class under the first ERM checkpoint, multiplied by class-dependent constants to address class imbalance.

6. Discussion

We have seen that we can profoundly simplify approaches to addressing spurious correlations while achieving state-of-the-art performance. AFR is driven by the key insight that, without either group labels or special intervention, a standard ERM model can be used to infer minority examples and then be minimally modified for improved group robustness. As a result, AFR is considerably simpler and faster than alternatives such as JTT and CNC, and more broadly applicable than methods like Group DRO and DFR which use group annotations during training. AFR reduces the barrier to deploying group robustness methods by reducing computational overhead, the need to recognize and label the spurious attributes a priori, and the need to use non-standard training procedures. Moreover, we have shown that AFR is robust to hyperparameters and can outperform methods like DFR that use group labels for training the model, when only a small number of group labels are available.

On the other hand, AFR is not without limitations. Similar to methods like JTT and CNC, AFR still benefits from some group-labeled data for hyperparameter tuning. As the number of groups increases, more group labels will be needed to estimate worst group performance and select the best hyperparameters, which may pose challenges for its use in datasets with numerous classes and spurious attributes. In addition, AFR cannot always re-balance the groups, though it achieves near-optimal balance without access to additional information. Addressing these limitations offers an exciting avenue for future research.

Acknowledgements

We thank Greg Benton and Yucen Lily Li for helpful discussions. This work is supported by NSF CAREER IIS-2145492, NSF I-DISRE 193471, NIH R01DA048764-01A1, NSF IIS-1910266, NSF 1922658 NRT-HDR, Meta Core Data Science, Google AI Research, BigHat Biosciences, Capital One, and an Amazon Research Award.

References

Borkan, D., Dixon, L., Sorensen, J., Thain, N., and Vasserman, L. Nuanced Metrics For Measuring Unintended Bias With Real Data For Text Classification. *Companion proceedings of the 2019 world wide web conference*, pp. 491–500, 2019.

Brendel, W. and Bethge, M. Approximating CNNs With Bag-of-local-Features Models Works Surprisingly Well On ImageNet. *arXiv preprint arXiv:1904.00760*, 2019.

Cui, Y., Jia, M., Lin, T.-Y., Song, Y., and Belongie, S. Class-balanced Loss Based On Effective Number Of Samples. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9268–9277, 2019.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training Of Deep Bidirectional Transformers For Language Understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F. A., and Brendel, W. ImageNet-trained CNNs Are Biased Towards Texture; Increasing Shape Bias Improves Accuracy And Robustness. *arXiv preprint arXiv:1811.12231*, 2018.

Geirhos, R., Jacobsen, J.-H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., and Wichmann, F. A. Shortcut Learning In Deep Neural Networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

He, K., Zhang, X., Ren, S., and Sun, J. Deep Residual Learning For Image Recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Idrissi, B. Y., Arjovsky, M., Pezeshki, M., and Lopez-Paz, D. Simple Data Balancing Achieves Competitive Worst-Group-Accuracy. *arXiv preprint arXiv:2110.14503*, 2021.

Irvin, J., Rajpurkar, P., Ko, M., Yu, Y., Ciurea-Ilcus, S., Chute, C., Henrik Marklund, B. H., Ball, R., Shpanskaya, K., Seekins, J., Mong, D. A., Halabi, S. S., Sandberg, J. K., Jones, R., Larson, D. B., Langlotz, C. P., Patel, B. N., Lungren, M. P., and Ng, A. Y. CheXpert: A Large Chest Radiograph Dataset With Uncertainty Labels And Expert Comparison. *Association for the Advancement of Artificial Intelligence (AAAI)*, 2019.

Izmailov, P., Kirichenko, P., Gruver, N., and Wilson, A. G. On Feature Learning in the Presence of Spurious Correlations. *Neural Information Processing Systems (NeurIPS)*, 35:38516–38532, 2022.

Kirichenko, P., Izmailov, P., and Wilson, A. G. Last Layer Re-Training Is Sufficient For Robustness To Spurious Correlations. *Preprint arXiv 2204.02937v1*, 2022.

Kish, L. *Survey Sampling*. John Wiley and Sons, New York, NY, 1965.

Koh, P. W., Sagawa, S., Marklund, H., Xie, S. M., Zhang, M., Balsubramani, A., Hu, W., Yasunaga, M., Phillips, R. L., Gao, I., et al. Wilds: A Benchmark Of In-The-Wild Distribution Shifts. *International Conference on Machine Learning (ICML)*, 2021.

Krizhevsky, A. and Hinton, G. Learning Multiple Layers Of Features From Tiny Images. *PhD Thesis*, 2009.

Krizhevsky, A., Hinton, G., et al. Learning Multiple Layers Of Features From Tiny Images. *University of Toronto (CS)*, 2009.

Lee, Y., Chen, A. S., Tajwar, F., Kumar, A., Yao, H., Liang, P., and Finn, C. Surgical Fine-Tuning Improves Adaptation To Distribution Shifts. *arXiv preprint arXiv:2210.11466*, 2022a.

Lee, Y., Yao, H., and Finn, C. Diversify And Disambiguate: Learning From Underspecified Data. *arXiv preprint arXiv:2202.03418*, 2022b.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

Lin, Y., Zhu, S., Tan, L., and Cui, P. ZIN: When and How to Learn Invariance Without Environment Partition? *Neural Information Processing Systems (NeurIPS)*, 2022.

Liu, E. Z., Haghgoo, B., Chen, A. S., Raghunathan, A., Koh, P. W., Sagawa, S., Liang, P., and Finn, C. Just Train Twice: Improving Group Robustness Without Training Group Information. *International Conference on Machine Learning (ICML)*, 2021.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep Learning Face Attributes In The Wild. *Proceedings of the IEEE international conference on computer vision*, 2015.

Menon, A. K., Rawat, A. S., and Kumar, S. Overparameterisation and Worst-Case Generalisation: Friend or Foe? *International Conference on Learning Representations (ICLR)*, 2020.

Moayeri, M., Pope, P., Balaji, Y., and Feizi, S. A Comprehensive Study Of Image Classification Model Sensitivity To Foregrounds, Backgrounds, And Visual Attributes. *arXiv preprint arXiv:2201.10766*, 2022.

Nam, J., Cha, H., Ahn, S., Lee, J., and Shin, J. Learning From Failure: De-biasing Classifier From Biased Classifier. *Neural Information Processing Systems (NeurIPS)*, 33:20673–20684, 2020.

Nam, J., Kim, J., Lee, J., and Shin, J. Spread Spurious Attribute: Improving Worst-group Accuracy With Spurious Attribute Estimation. *International Conference on Learning Representations*, 2022.

Oakden-Rayner, L., Dunnmon, J., Carneiro, G., and Ré, C. Hidden Stratification Causes Clinically Meaningful Failures In Machine Learning For Medical Imaging. *Proceedings of the ACM conference on health, inference, and learning*, pp. 151–159, 2020.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. ImageNet Large Scale Visual Recognition Challenge. *International journal of computer vision*, 115 (3):211–252, 2015.

Sagawa, S., Koh, P. W., Hashimoto, T. B., and Liang, P. Distributionally Robust Neural Networks For Group Shifts: On The Importance Of Regularization For Worst-Case Generalization. *International Conference on Learning Representations (ICLR)*, 2020.

Sohoni, N., Dunnmon, J., Angus, G., Gu, A., and Ré, C. No Subclass Left Behind: Fine-Grained Robustness In Coarse-Grained Classification Problems. *Neural Information Processing Systems (NeurIPS)*, 33:19339–19352, 2020.

Sohoni, N., Sanjabi, M., Ballas, N., Grover, A., Nie, S., Firooz, H., and Ré, C. BARACK: Partially Supervised Group Robustness With Guarantees. *arXiv preprint arXiv:2201.00072*, 2021.

Taghanaki, S. A., Choi, K., Khasahmadi, A., and Goyal, A. Robust Representation Learning Via Perceptual Similarity Metrics. *International Conference on Machine Learning (ICML)*, 2021.

Wang, X., Peng, Y., Lu, L., Lu, Z., Bagheri, M., and Summers, R. M. Chestx-ray8: Hospital-scale Chest X-Ray Database And Benchmarks On Weakly-Supervised Classification And Localization Of Common Thorax Diseases. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2097–2106, 2017.

Williams, A., Nangia, N., and Bowman, S. R. A Broad-Coverage Challenge Corpus For Sentence Understanding Through Inference. *arXiv preprint arXiv:1704.05426*, 2017.

Xiao, K., Engstrom, L., Ilyas, A., and Madry, A. Noise Or Signal: The Role Of Image Backgrounds In Object Recognition. *arXiv preprint arXiv:2006.09994*, 2020.

Yang, W., Kirichenko, P., Goldblum, M., and Wilson, A. G. Chroma-VAE: Mitigating Shortcut Learning With Generative Classifiers. *Neural Information Processing Systems (NeurIPS)*, 2022.

Yang, Y., Zhang, H., Katabi, D., and Ghassemi, M. Change is Hard: A Closer Look at Subpopulation Shift. *arXiv preprint arXiv:2302.12254*, 2023.

Zech, J. R., Badgeley, M. A., Liu, M., Costa, A. B., Titano, J. J., and Oermann, E. K. Variable Generalization Performance Of A Deep Learning Model To Detect Pneumonia In Chest Radiographs: A Cross-Sectional Study. *PLoS medicine*, 15(11):e1002683, 2018.

Zhang, M., Sohoni, N. S., Zhang, H. R., Finn, C., and Ré, C. Correct-n-Contrast: A Contrastive Approach For Improving Robustness To Spurious Correlations. *Preprint arXiv 2203.01517v1*, 2022.

Appendix Outline

This Appendix is organized as follows:

- In Section A, we provide code for computing the weights in AFR.
- In Section B, we describe the details of the experimets.
- In Section C, we provide additional ablations.
- In Section D, we discuss additional related work.

A. Automatic Feature Reweighting Implementation Details

In this section, we provide the code for computing the weights for the weighted loss in AFR.

```
def compute_afr_weights(erm_logits, class_label, gamma):
  # erm_logits: (n_samples, n_classes)
  # class_label: (n_samples,)
  # gamma: float
  with torch.no_grad():
    p = erm_logits.softmax(-1)
    y_onehot = torch.zeros_like(erm_logits).scatter_(-1, class_label.unsqueeze(-1), 1)
    p_true = (p * y_onehot).sum(-1)
    weights = (-gamma * p_true).exp()
    n_classes = torch.unique(class_label).numel()
    # class balancing
    class_count = []
    for y in range(n_classes):
      class_count.append((class_label == y).sum())
    for y in range(1, n_classes):
      weights[class_label == y] *= class_count[0] / class_count[y]
    weights /= weights.sum()
  return weights
```

B. Experimental details

B.1. Last layer retraining implementation

To retrain the last layer, we pre-compute and store the embeddings produced by the stage 1 ERM model. Unless stated otherwise, we use full-batch gradient descent without momentum to retrain the last layer parameters, initializing them to their values at the end of stage 1. We clip the ℓ_2 norm of the gradient vector to 1. The epoch achieving highest validation worst group accuracy is used to report our result, except for the experiment in Section 4.4 where we do not perform early stopping on Waterbirds and CelebA when not using the entire validation set, which we found to improve performance by avoiding stopping too early.

B.2. Timing experiment details

Timings for AFR and JTT in Figure 3 are obtained by running on a single RTX8000 (48 GB) NVIDIA GPU. For JTT we have the following (epochs, batch size) tuples per dataset which incorporate the two stages of training. Waterbirds (360, 64), CelebA (51, 128), MultiNLI (7, 32), CivilComments (7, 16), and CXR (150, 128). For CNC, we present timings reported by Zhang et al. (2022).

B.3. Dataset details

We now detail the models and hyperparameters used on each of the tasks.

- **Waterbirds:** ResNet-50 (torchvision.models.resnet50(pretrained=True))

  - **AFR** 1st stage: epochs = 50, optimizer=sgd, scheduler=cosine, batch size = 32, learning rate = 3e-3, weight decay = 1e-4.
  - **AFR** 2nd stage: epochs = 500, γ from 33 points linearly spaced between $[4, 20]$, learning rate in = 1e-2, $\lambda \in \{0, 0.1, 0.2, 0.3, 0.4\}$.
  - **JTT** same hyperparameters as in Liu et al. (2021).
  - **CNC** same hyperparameters as in Zhang et al. (2022).

- **CelebA** ResNet-50 (torchvision.models.resnet50(pretrained=True))
  - **AFR** 1st stage: epochs = 20, optimizer=sgd, scheduler=cosine, batch size = 32, learning rate = 3e-3, weight decay = 1e-4.
  - **AFR** 2nd stage: epochs = 1000, γ from 10 points linearly spaced between $[1, 3]$, learning rate =2e-2, $\lambda \in \{0.001, 0.01, 0.1\}$.
  - **JTT** same hyperparameters as in Liu et al. (2021).
  - **CNC** same hyperparameters as in Zhang et al. (2022).

- **MultiNLI** BERT (using HuggingFace)
  - **AFR** 1st stage: epochs = 5, optimizer = SGD, scheduler = constant, batch size = 32, learning rate = 2e-5, weight decay = 0.
  - **AFR** 2nd stage: epochs = 200, γ from 10 points linearly spaced between $[1e2, 1e5]$, learning rate = 1e-2, λ from 26 points linearly spaced between $[0, 50]$.
  - **JTT** same hyperparameters as in Liu et al. (2021).
  - **CNC** same hyperparameters as in Zhang et al. (2022).

- **CivilComments** BERT (using HuggingFace)
  - **AFR** 1st stage: epochs = 3, optimizer = SGD, scheduler = constant, batch size = 24, learning rate = 2e-5, weight decay = 0.
  - **AFR** 2nd stage: $\gamma \in \{0, 0.01, 0.1, 1, 3, 10\}$, $\lambda = 0$, learning rate $3 \cdot 10^{-3}$. For hyperparameter tuning and early stopping, we combine all religions into a single attribute, as we found it to produce more stable results.
  - **JTT** same hyperparameters as in Liu et al. (2021).
  - **CNC** same hyperparameters as in Zhang et al. (2022).

- **Chest X-Ray** DenseNet121 (torchvision.models.densenet121(pretrained=True))
  - **AFR** 1st stage: epochs = 100, optimizer = AdamW, scheduler = constant, batch size = 32, learning rate = 1e-4, weight decay = 1e-5.
  - **AFR** 2nd stage: $\gamma \in [1, 6]$, $\lambda = 0$, learning rate $\in \{10^{-4}, 50^{-4}, 10^{-3}, 50^{-3}, 10^{-2}, 50^{-2}\}$.
  - **JTT** same hyperparameters as in Yang et al. (2022).

- **CIFAR-10** ResNet-18 (torchvision.models.resnet18(pretrained=True))
  - (Krizhevsky & Hinton, 2009), is an image classification task with no obvious spurious features. we want to identify to what of 10 groups (airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck) does an image belong to.
  - **AFR** 1st stage / **ERM**: epochs = 20, optimizer = AdamW, scheduler = cosine, batch size = 32, learning rate = 1e-4, weight decay = 5e-4.
  - **AFR** 2nd stage: epochs = 500, optimizer = sgd, scheduler = constant, batch size = full batch, $\gamma \in \{1, 2, 5, 10\}$, learning rate in $\{1e-4, 1e-3, 1e-2\}$, weight decay = 0. $\lambda = 0$.

- **Camelyon17** DenseNet-121 (torchvision.models.densenet121(pretrained=True))
  - (Zech et al., 2018; Koh et al., 2021) is an image classification task where the goal is to predict whether a a digital whole-slide image (WSIs) of a lymph node section contains cancer metastases.


(a) Waterbird on water


(b) Landbird on water

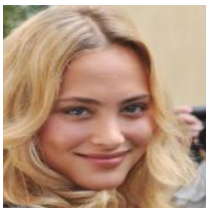

(c) Waterbird on land


(d) Lanbird on land

Figure 8: **Example images on Waterbirds.** The objective is to identify the bird type (landbird $y = 0$ vs waterbird $y = 1$) where the spurious feature is the background (water vs land). The groups are $\mathcal{G}_1 =$ lanbird on land, $\mathcal{G}_2 =$ lanbird on water, $\mathcal{G}_3 =$ waterbird on land, and $\mathcal{G}_4 =$ waterbird on water. In terms of number of training samples per group we have $\mathcal{G}_1 = 3,498(73\%)$, $\mathcal{G}_2 = 184(4\%)$, $\mathcal{G}_3 = 56(1\%)$, and $\mathcal{G}_4 = 1,057(22\%)$.


(a) Non-blond female


(b) Non-blond male


(c) Blond female


(d) Blond male

Figure 9: **Example images on CelebA.** The objective is to differentiate hair color (non-blond $y = 0$ vs blond $y = 1$) where the spurious feature is the gender (female vs male). The groups are $\mathcal{G}_1 =$ non-blond female, $\mathcal{G}_2 =$ non-blond male, $\mathcal{G}_3 =$ blond female, and $\mathcal{G}_4 =$ blond male. In terms of number of training samples per group we have $\mathcal{G}_1 = 71,629(44\%)$, $\mathcal{G}_2 = 66,874(41\%)$, $\mathcal{G}_3 = 22,880(14\%)$, and $\mathcal{G}_4 = 1,387(1\%)$.

- **AFR** 1st stage / **ERM**: epochs = 20, optimizer = AdamW, scheduler = constant, batch size = 256, learning rate = 1e-3, weight decay = 1e-4. When evaluating the final worst group accuracy, we treat all groups separately, as prescribed in Koh et al. (2021).
- **AFR** 2nd stage: epochs = 500, optimizer = SGD, scheduler = constant, batch size = full batch, $\gamma \in (1, 100)$, learning rate in $\{1e-4, 5e-4, 1e-3, 5e-3, 1e-2\}$, weight decay = 0. $\lambda = 0$.

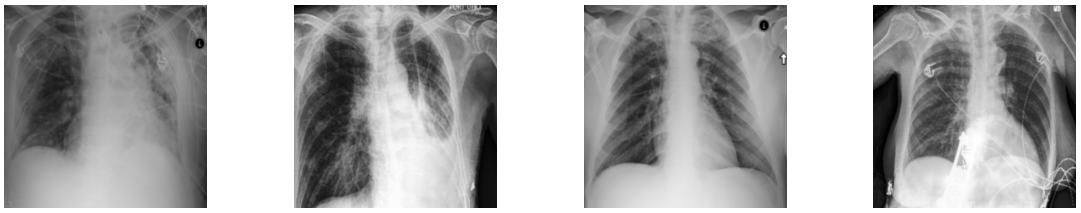
C. Additional Ablations

C.1. Datasets without known spurious features

We evaluate AFR on the CIFAR-10 (Krizhevsky et al., 2009) and Camelyon17 (Zech et al., 2018; Koh et al., 2021) datasets where there are no clearly identified spurious features that we can label. For CIFAR-10 we compute the test worst class accuracy and test mean accuracy. For CIFAR-10 we use ResNet-18 (He et al., 2016) pre-trained on ImageNet1k and for Camelyon17 we use ResNet-50 pretrained on ImageNet1k. As Camelyon17 is a domain generalization benchmark, we compute the test mean accuracy on hospital 2 and hospital 4, which are not represented in the training data (hospitals 1, 3, and 5). Reassuringly, we find that AFR only marginally affects the performance on all evaluations compared to ERM, suggesting that it can be safely applied even in situations when we are unsure if the data contains spurious features.

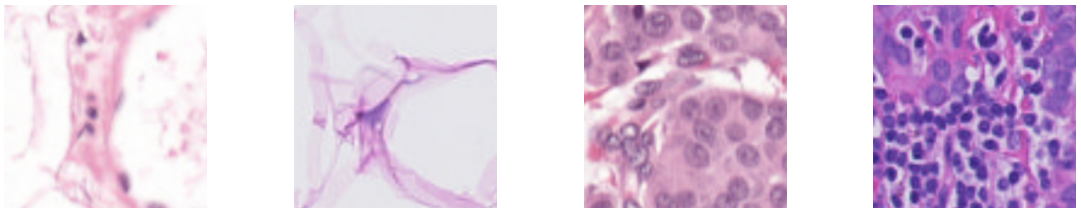
C.2. Sensitivity to γ on CelebA

Similar to what we observed on Waterbirds, Figure 14 shows AFR performs well across a wide range of γ. We set $\lambda = 0$ for simplicity. On both CelebA and Waterbirds, AFR can be applied to improve worst group accuracy without tuning either γ or λ.



(a) NIH - pneumonia (b) CheXpert - pneumonia (c) NIH - healthy (d) CheXpert - healthy

Figure 10: **Example images on Chest-X-Ray**. The objective is to identify if the patient has pneumonia (healthy $y = 0$ vs pneumonia $y = 1$) where the spurious features are the hospital markings from the different sources (NIH and CheXpert). The groups are $\mathcal{G}_1 =$ NIH - no, $\mathcal{G}_2 =$ NIH - pneumonia, $\mathcal{G}_3 =$ CheXpert - no, and $\mathcal{G}_4 =$ CheXpert - pneumonia. In terms of number of training samples per group we have $\mathcal{G}_1 = 9,281(47\%)$, $\mathcal{G}_2 = 978(5\%)$, $\mathcal{G}_3 = 8,766(44\%)$, and $\mathcal{G}_4 = 855(4\%)$.



(a) H123 - No (b) H5 - No (c) H123 - Cancer (d) H5 - Cancer

Figure 11: **Example images on Camelyon-17.** The objective is to identify if a lymph node slide has cancer (no cancer $y = 0$ vs metastases $y = 1$) where the spurious features are the hospital slide characteristics. In this case, the in-distribution sources are hospitals 1, 2, and 3 (H123) and the out-of-distribution source is hospital 5 (H5). The groups are $\mathcal{G}_1 =$ H123 - no, $\mathcal{G}_2 =$ H4 - no, $\mathcal{G}_3 =$ H123 - Cancer, and $\mathcal{G}_4 =$ H4 - Cancer. In terms of number of data samples per group we have $\mathcal{G}_1 = 151,390(39\%)$, $\mathcal{G}_2 = 42,527(11\%)$, $\mathcal{G}_3 = 151,046(38\%)$, and $\mathcal{G}_4 = 42,627(12\%)$. In contrast to other datasets, $\mathcal{G}_2$ and $\mathcal{G}_4$ are not observed in the training data.

airplane car bird cat deer dog frog horse ship truck

Figure 12: **Example images on CIFAR-10.** CIFAR-10 consist of tiny images of 10 different categories: airplane ($y = 0$), automobile ($y = 1$), bird ($y = 2$), cat ($y = 3$), deer ($y = 4$), dog ($y = 5$), frog ($y = 6$), horse ($y = 7$), ship ($y = 8$) and truck ($y = 9$). The train size if 50K and each class consists of 8.3K images.

MultiNLI

	Text examples	Class label	Description	# Train data
$\mathcal{G}_1$	*"if residents are unhappy, they can put wheels on their homes and go someplace else, she said. [SEP] residents are stuck here but they can't go anywhere else."*	0	contradiction no negations	57498 (28%)
$\mathcal{G}_2$	*"within this conflict of values is a clash about art. [SEP] there is no clash about art."*	0	contradiction has negations	11158 (5%)
$\mathcal{G}_3$	*"there was something like amusement in the old man's voice. [SEP] the old man showed amusement."*	1	entailment no negations	67376 (32%)
$\mathcal{G}_4$	*"in 1988, the total cost for the postal service was about $36. [SEP] the postal service cost us citizens almost nothing in the late 80's. "*	1	entailment has negations	1521 (1%)
$\mathcal{G}_5$	*"yeah but even even cooking over an open fire is a little more fun isn't it [SEP] i like the flavour of the food."*	2	neutral no negations	66630 (32%)
$\mathcal{G}_6$	*"that's not too bad [SEP] it's better than nothing"*	2	neutral has negations	1992 (1%)

Target: contradiction / entailment / neutral; **Spurious feature**: has negation words. **Minority**: $\mathcal{G}_4$, $\mathcal{G}_6$

CivilComments

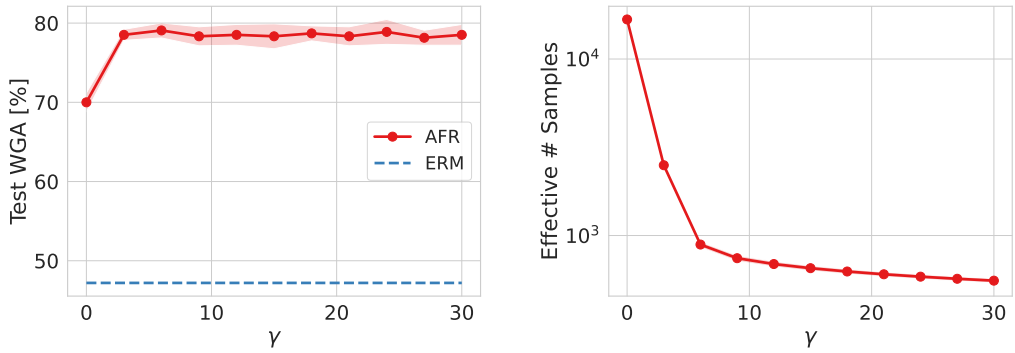
Text examples	Class label	Description	# Train data
"I'm quite surprised this worked for you. Infrared rays cannot penetrate tinfoil."	0	non-toxic no identities	148186 (55%)
"I think you may have misunderstood what 'straw men' are. But I'm glad that your gravy is good."	0	non-toxic has identities	90337 (33%)
"Hahahaha putting his faith in Snopes. Pathetic."	1	toxic no identities	12731 (5%)
"That sounds like something a white person would say."'	1	toxic has identities	17784 (7%)

Target: Toxic / not toxic comment; **Spurious feature**: mentions protected categories.

Figure 13: Dataset examples for MultiNLI and CivilComments. We underline the words corresponding to the spurious feature. CivilComments contains 16 overlapping groups corresponding to toxic / non-toxic comments and mentions of one of the protected identities: male, female, LGBT, black, white, Christian, Muslim, other religion. We only show examples with mentions of the male and white identities.

Method	CIFAR-10		Camelyon17	
	Worst(%)	Mean(%)	H123 Mean(%)	H5 Mean(%)
ERM	89.6	95.1	85.0	90.1
AFR	$89.3_{\pm 0.5}$	$94.9_{\pm 0.1}$	$87.1_{\pm 0.1}$	$88.9_{\pm 0.4}$

Table 2: **Performance on data with unidentified spurious features.** For CIFAR-10 we report test worst class accuracy and mean accuracy. For Camelyon17 we report mean accuracy for two different hospital's data not seen during training. H123 relates to data from hospitals 1, 2, and 3 used for training and H5 to the data from hospital 5 which is out of distribution. When spurious correlations are not identified, AFR recovers the standard ERM performance.



(a) Test WGA

(b) Effective sample size

Figure 14: **AFR's γ robustness on CelebA. (a)** AFR achieves high test worst group accuracy on CelebA across a wide range of γ. For simplicity, we do not tune the regularization parameter λ and set it to 0. We show mean and standard deviation across 3 runs. **(b)** The effective sample size decreases and then stabilizes as γ increase.

C.3. Ablating early stopping and ℓ_2 regularization

Similar to JTT and CNC, AFR employs early stopping during the 2nd stage. We re-ran AFR without early stopping while allowing the other hyperparameters (γ and λ) to be tuned as usual. As expected, AFR's performance decreases, but we still observe substantial improvement over ERM, showing that 2nd stage early stopping helps but is not essential to AFR's performance gain. Similarly, we found ablating ℓ_2 regularization (setting $\lambda = 0$ and only tuning γ) only slightly degrades AFR's performance. These findings show AFR's performance gain does not rely on carefully tuned regularization. Table 4 summarizes these results.

Method	Waterbirds	MultiNLI	Chest X-Ray
AFR	0.022	0.07	0.10

Table 3: **P-values** testing the null hypothesis that AFR's mean test WGA is lower than 2nd best performing method. When the 2nd best performing method is CnC, we perform a two-sample t-test; when it's JTT, we perform a one-sample t-test since only its mean was reported

Method	Waterbirds	CelebA	MultiNLI
ERM	72.6	47.2	67.9
AFR	$90.4_{\pm 0.1}$	$82.0_{\pm 0.5}$	$73.4_{\pm 0.6}$
AFR w/o ℓ_2 reg.	$90.1_{\pm 0.7}$	$81.3_{\pm 1.8}$	$71.1_{\pm 1.4}$
AFR w/o early stopping	$89.9_{\pm 0.7}$	$80.0_{\pm 1.6}$	$71.1_{\pm 1.4}$

Table 4: **AFR's performance without early stopping or ℓ_2 regularization**. AFR's performance gain does not rely on carefully tuned regularization. Reported mean and standard deviation are computed over three independent runs.

C.4. Improving group robustness with no hyperparameter tuning

To show that AFR improves group robustness without carefully tuning γ, ℓ_2 regularization, or early stopping, we run AFR on Waterbirds and CelebA where we set γ to four positive values $\{1, 2, 4, 8\}$ and run it for 100 steps without ℓ_2 regularization. We found in most cases AFR still significantly improves test WGA compared to ERM. While the optimal value for γ depends on the dataset, most values between 1 and 8 lead to substantial improvement over ERM with the exception of $\gamma = 1$ on Waterbirds where the value appears to be too low for AFR upweight the minority group examples.

	Waterbirds	**CelebA**
$\gamma = 1, \lambda = 0$, 100 steps	$72.3_{\pm 1.8}$	$75.9_{\pm 1.7}$
$\gamma = 2, \lambda = 0$, 100 steps	$76.2_{\pm 3.5}$	$81.1_{\pm 1.4}$
$\gamma = 4, \lambda = 0$, 100 steps	$84.4_{\pm 3.3}$	$72.1_{\pm 1.1}$
$\gamma = 8, \lambda = 0$, 100 steps	$84.1_{\pm 4.7}$	$61.0_{\pm 0.9}$
ERM	72.6	47.2

Table 5: **AFR's outperforms ERM without hyperparameter tuning.** AFR improves WGA on Waterbirds and CelebA without tuning hyperparameters at all. Reported mean and standard deviation are computed over three independent runs.

C.5. Robustness to specification of the weights

While simple and intuitive, the choice to define the weights as $\mu_i \propto \beta_{y_i} \exp(-\gamma\, \hat{p}_i)$ is somewhat arbitrary. However, as the goal is to simply upweight the poorly predicted examples, we again do not expect the performance of AFR to be sensitive to the exact functional form used for μ_i, as long as it is large for poorly predicted examples. We verify this by replacing $\exp(-\gamma\, \hat{p}_i)$ with two alternatives $\hat{p}_i^{-\gamma} = \exp(\gamma(-\log \hat{p}_i))$ and $(1 - \hat{p}_i)^\gamma$, as used by focal loss[1] (Lin et al., 2017), and compare the resulting test WGA after tuning γ on validation WGA in Table 6. Indeed, both variants of AFR are able to drastically improve test WGA compared to ERM and achieve similar performance as the original one.

Method	**Test WGA**
ERM	72.6
AFR $\exp(-\gamma\, \hat{p}_i)$	$\mathbf{90.4}_{\pm \mathbf{1.1}}$
AFR $(1 - \hat{p}_i)^\gamma$	$89.3_{\pm 1.9}$
AFR $\hat{p}_i^{-\gamma}$	$89.1_{\pm 1.3}$

Table 6: **Performance on alternative functional choices.** Test WGA achieved on the Waterbirds by ERM and variants of AFR dataset using alternative definitions for the weights μ_i. γ is always chosen to maximize validation worst group accuracy. The performance of AFR is not sensitive to the choice of the exact functional form for the weights as long as it upweights poorly predicted examples. The mean and standard deviation are computed over 3 independent runs.

C.6. Robustness to splitting ratio

The $80\% : 20\%$ splitting ratio between $\mathcal{D}_{\text{ERM}}$ and $\mathcal{D}_{\text{RW}}$ is intended to keep most of training data for learning a sufficiently good feature extractor and only a small portion for last layer retraining since the latter tends to be much more sample-efficient. To study the sensitivity of AFR's performance to this ratio, we compare the default choice with three alternatives. Table 7 shows the default choice of $80\% : 20\%$ performs well and the performance is not very sensitive to the value of this ratio. For all four choices, AFR significantly improves over ERM on both datasets.

A clear case where AFR would fail to improve group robustness is when $\mathcal{D}_{\text{RW}}$ does not contain any minority group examples to upweight. Luckily, this scenario is extremely unlikely when $\mathcal{D}_{\text{RW}}$ is drawn from the train distribution. For instance, suppose that we are considering a typical deep learning task where the training set has $\geq 10k$ examples. Moreover, lets suppose the minority group constitutes only 5% of the training data. Then there are on average 100 minority group examples in the 2nd split, with a standard deviation of around 30. The probability of having fewer than 10 minority group examples (a 3σ event) is less than 0.2%. Therefore, in practice there will be minority group examples, so long as the user follows our

[1] However, unlike in focal loss, our weights are fixed throughout training and do not depend on the parameters being optimized.

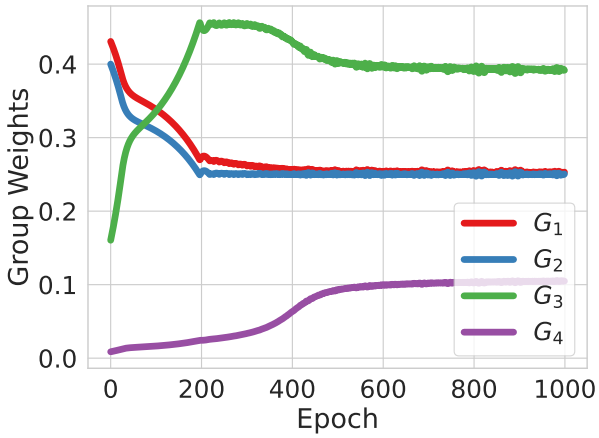


Figure 15: **A trained neural network fails to completely balance group weights on CelebA.** Group aggregated weights on CelebA produced by a neural network trained to minimize group-imbalance.

recommendations for the split.

$\lvert\mathcal{D}_{\text{ERM}}\rvert : \lvert\mathcal{D}_{\text{RW}}\rvert$	**Waterbirds**	**CelebA**
50:50	$88.8_{\pm 1.0}$	$80.9_{\pm 0.3}$
70:30	$89.8_{\pm 1.1}$	$84.2_{\pm 1.6}$
80:20	$90.4_{\pm 1.1}$	$82.0_{\pm 0.5}$
90:10	$89.0_{\pm 2.7}$	$82.7_{\pm 0.2}$
ERM	72.6	47.2

Table 7: **AFR is robust to the splitting ratio.** Test WGA achieved on CelebA by AFR with different splitting ratios. The default value of $80\% : 20\%$ performs well, but the performance is not very sensitive to this value. For all four choices, AFR significantly improves over ERM. The mean and standard deviation are computed over 3 independent runs

C.7. Impossibility for group-balanced weights on CelebA

In general, it is not possible to define weights μ_i purely as a function of the output of an ERM model trained on $\mathcal{D}_{\text{ERM}}$ such that they are balanced over the groups on the held-out dataset $\mathcal{D}_{\text{RW}}$. To verify this claim, we train a neural network f to minimize the following loss

$$\mathcal{L} = \frac{1}{4}\sum_{g=1}^{4}\left|\left(\sum_{i\in\mathcal{D}_{\text{RW}}:g_i=g} f(p_i^{\text{ERM}}, y_i)\right) - \frac{1}{4}\right|,$$

which measures average deviation from $1/4$ of the group aggregated weights produced by the network over 4 groups. The input to f is (p_i^{ERM}, y_i), the predicted probabilities for each class by the first-stage ERM model, and the true class label y_i. We use an MLP with two hidden layers and 128 units each, whose output is constrained to be positive using a softplus function and normalized to sum to unity over all examples. Figure 15 shows the group aggregated weights on CelebA produced by the network during training, using the Adam optimizer. The group aggregated weights don't converge to $1/4$, showing that any upweighting strategy based only on ERM prediction and class label is unlikely to produce group-balanced weights on CelebA.

D. Additional Related Work

On spurious correlations. Numerous studies expose how neural networks rely on spurious correlations for a diverse set of real word problems. In image classification, neural networks rely on the background of the image and not on the actual objects as seen in Xiao et al. (2020), Sagawa et al. (2020) and Moayeri et al. (2022). Additionally, neural network classifiers

might rely on object textures (Geirhos et al., 2018) or on small image features (not actual spatial relationships) as argued in Brendel & Bethge (2019). Critically, for chest X-ray classification tasks, neural networks have been shown to rely on hospital specific tokens, chest drains or other features that irrelevant to the diagnosis of pneumonia as reported in Zech et al. (2018) or Oakden-Rayner et al. (2020). For a comprehensive survey of the area, see Geirhos et al. (2020), and Yang et al. (2023) provide a comprehensive evaluation of the existing methods.

Leveraging group annotations. When group annotations are present there are several methods that can provide high WGA. We can leverage the group annotation through: (i) class or group balancing or weighting (Cui et al., 2019; Menon et al., 2020; Idrissi et al., 2021; Kirichenko et al., 2022; Izmailov et al., 2022), or via (ii) distributionally robust optimization (Sagawa et al., 2020), or finally via (iii) contrastive methods (Taghanaki et al., 2021). Taghanaki et al. (2021) propose CIM: a method that leverages a contrastive loss and pixel-level image statistics to learn input-space transformations that improve performance on downstream tasks. However, requiring annotations for a large dataset can be expensive. Worse, as we showed in Section 4.2, there are problems where it is not evident what the spurious feature is. Additionally there could be several spurious features present at the same time creating more difficulties.

Annotation-free methods. There has been plenty of efforts on developing methods that achieve high WGA without requiring group annotations. A common theme amongst these methods is the presence of two stages: first, train a checkpoint using ERM and then, modify this model to improve WGA. We now focus on the methods that were not discussed in Section 2. In GEORGE (Sohoni et al., 2020) the authors infer group annotations based on the clusters formed in the feature space learnt by the ERM feature extractor $f_{\hat{\phi}}$. With these discovered groups, the authors then run GDRO. In Nam et al. (2022) and Sohoni et al. (2021), the authors use semi-supervised learning to propagate the limited available group labels to the entire dataset. In LfF (Nam et al., 2020) the authors use a generalization of the cross-entropy loss as to identify samples that have a strong agreement between the output of the neural network and the label and then to score higher the samples that do not. Then, having identified the "difficult" samples, the authors upweight the cross-entropy loss using the score. Finally, as shown in Section 2, JTT (Liu et al., 2021) and CNC (Zhang et al., 2022) are also methods that improve WGA without requiring group annotations.

In Lin et al. (2022), the authors argue how in general it is impossible to perfectly separate the spurious and invariant features without auxiliary information. We note that the good performance of AFR does not contradict their finding. First, in our experiments we use the auxiliary information about the nature of the spurious features, as we tune the hyperparameters of the method for worst group accuracy on the validation set. More importantly, AFR does not aim to learn invariant features, rather it uses the same features learned by a standard ERM model and only retrains the last layer on a weighted heldout dataset that upweights minority group examples. The discovery that such a procedure is sufficient to achieve SOTA worst group performance on various vision and text benchmarks is an interesting and important one, which does not contradict findings in Lin et al. (2022).